CS 4250 - Course Project Requirements

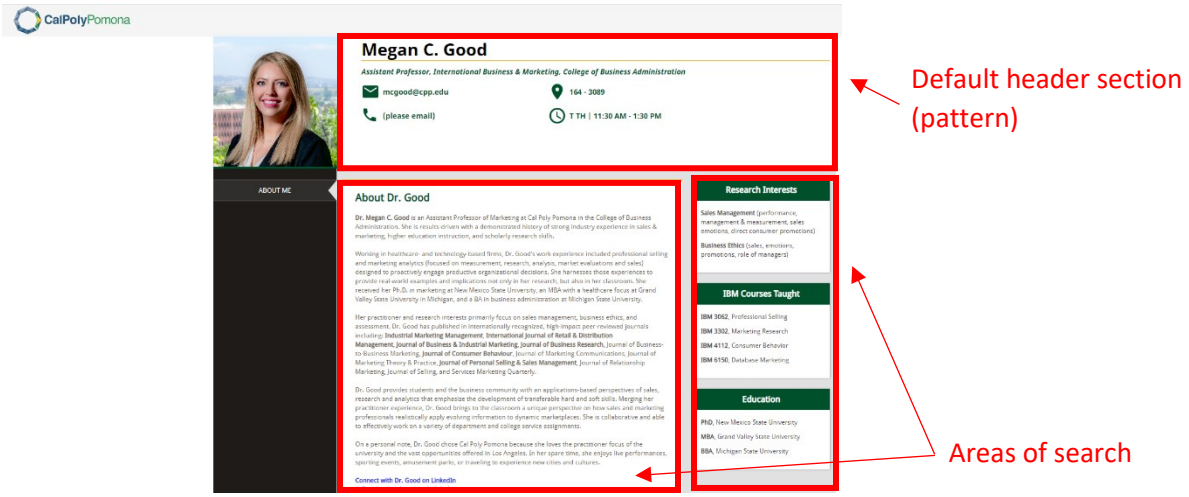
CPP wants to build a search engine for students to search and explore faculty research work. Currently, this task is usually conducted manually through the professors Web pages, considerably slowing down the search process. Three departments will be used now as a pilot: Biology, Civil Engineering, International Business and Marketing. See their department site below.

- **Biology Department (10 professors to be found).**
  - **Seed URL: https://www.cpp.edu/sci/biological-sciences/index.shtml**

- **Civil Engineering Department (minimum of 10 professors).**
  - **Seed URL: https://www.cpp.edu/engineering/ce/index.shtml**

- **International Business and Marketing Department (minimum of 10 professors).**
  - **Seed URL: https://www.cpp.edu/cba/international-business-marketing/index.shtml**

Based on the problem specified, your team should build a topical search engine that will crawl the three corresponding websites (see the URL informed) until all individual professor web pages are found (see an example from the International Business and Marketing department below – Dr. Megan Good page). Those pages will include a default header section which will work as a pattern for your crawler - see below. Next, this information should be collected, transformed, and saved in a MongoDB to enabling searching.



Faculty CPP page template

Requirements:

1) To write this search engine, first strictly follow the pseudocode shown below to crawl the data. This pseudocode works for each individual department, so you need to make the appropriate adjustments to crawl data from all three departments. Your frontier must include at the beginning only the single seed URLs informed before (**department home page**) and from these pages, search through all linked pages until the target pages are found. Links might appear with full or relative addresses, and your crawler needs to consider this. Broken links might also be found.
2) Stop criteria: when the crawler finds all Faculty pages that follow the template (see image above).
3) Use the Python libraries `urllib`, `BeautifulSoup`, and `PyMongo`.
4) Use the `MongoDB` collection `pages` to persist pages' data.

```
procedure crawlerThread (frontier, num_targets)
  targets_found = 0
  while not frontier.done() do
     url <- frontier.nextURL()
     html <- retrieveURL(url)
     storePage(url, html)
     if target_page (parse (html))
        targets_found = targets_found + 1
     if targets_found = num_targets
        clear_frontier()
     else
       for each not visited url in parse (html) do
           frontier.addURL(url)
       end for
  end while
end procedure
```

5) After crawling the department data, including faculty data, your search engine should parse and index only the latter – more specifically, only the area of search content (look the Faculty CPP page template image on the previous page). Use the Python libraries `BeautifulSoup` and `PyMongo`.
6) Finally, your search engine should allow a user to enter an arbitrary query, and based on that, retrieve the faculty pages that are relevant to the query. For each returned document, include its link and text snipped, providing pagination to show only 5 clickable documents per page. Use the Python library `scikit-learn`.

Processes that should not be addressed: GUI building and authentication.